ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

550 Jahre
Albert-Ludwigs-
Universität Freiburg
1457 – 2007

Humboldt Universität zu Berlin
18.2.2008 Berlin-Adlershof

Emulation - Eine Archivierungsstrategie zur Erhaltung des Langzeitzugriffs auf digitale dynamische Objekte

Dirk von Suchodoletz
Lehrstuhl für Kommunikationssysteme
Prof. Schneider

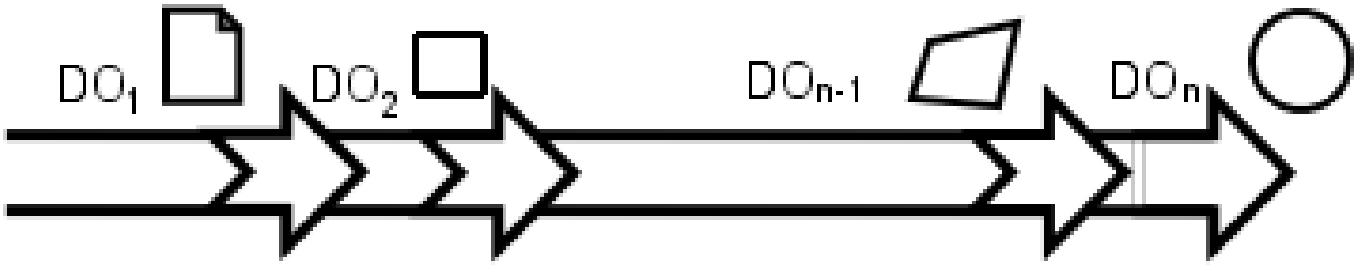
Rechenzentrum der Universität Freiburg

Klassische digitale Objekte I

- Texte und allgemeine Dokumente in verschiedenen möglichst wohldefinierten Formaten
- Audio-, Videodatenströme
- Typischerweise wird “Viewer” benötigt
  - entweder gebunden an eine Applikation
  - oder standardisiertes Format, welches Auswahl erlaubt

Klassische digitale Objekte II

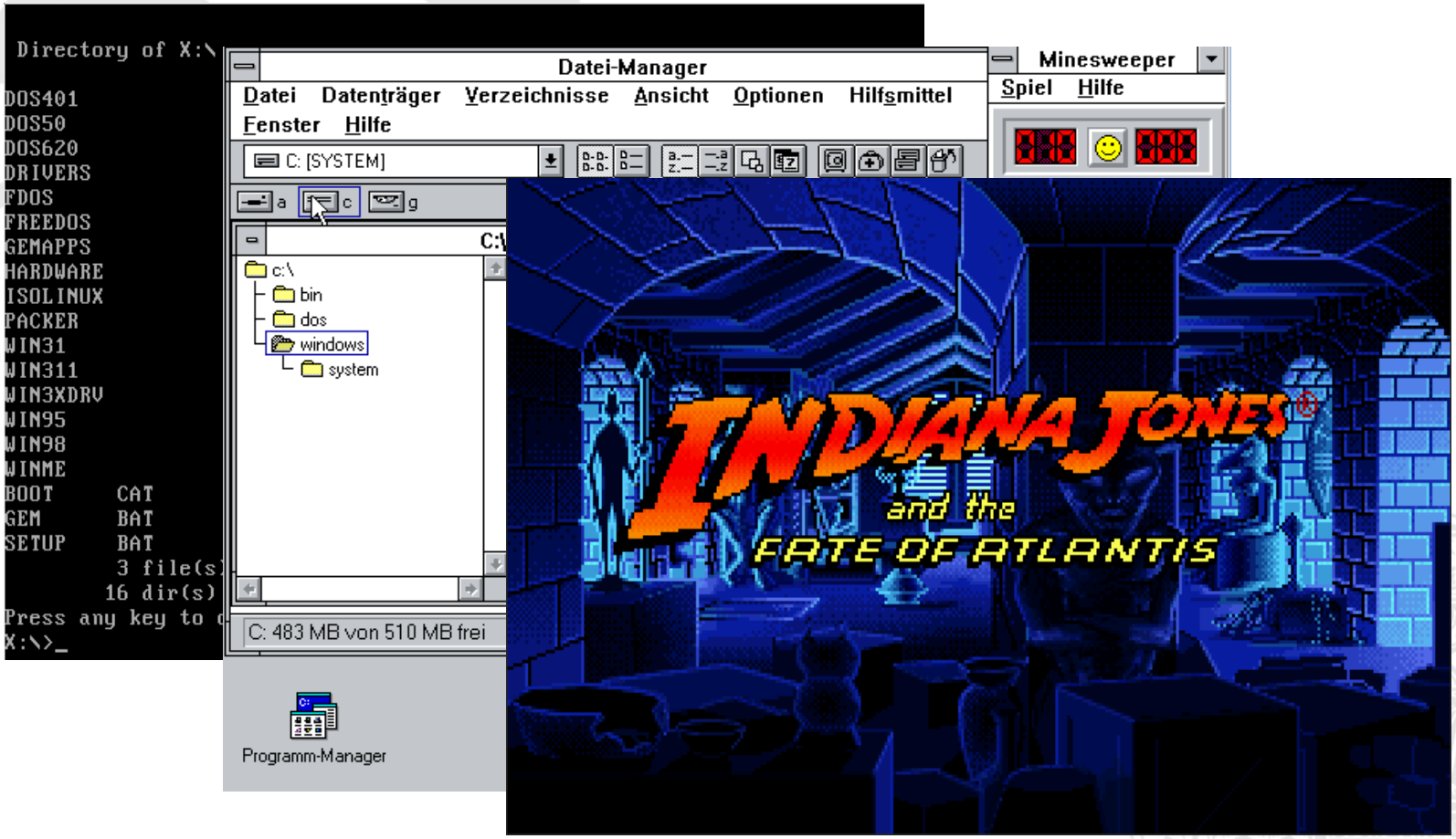
- Zentrale Eigenschaften
  - statisch – keine User-Interaktion
  - geeignet für Migration
    - automatisierbar, jedoch nicht ganz unproblematisch – Authentizitätsproblem des n-fach migrierten Objekts



Weitere digitale Objekte

- Darüber hinaus große Anzahl weiterer *dynamischer* digitaler Objekte
  - Objekte benötigen passende Laufzeitumgebung etliche davon *interaktiv* (Benutzerinteraktion)
    - Computerprogramme jedwelcher Art
    - Betriebssysteme
    - Digitale Enzyklopädien, E-Learning-Umgebungen
    - Datenbanken, Content Management Systeme
    - Multimediaapplikationen, Spiele

Dynamische digitale Objekte

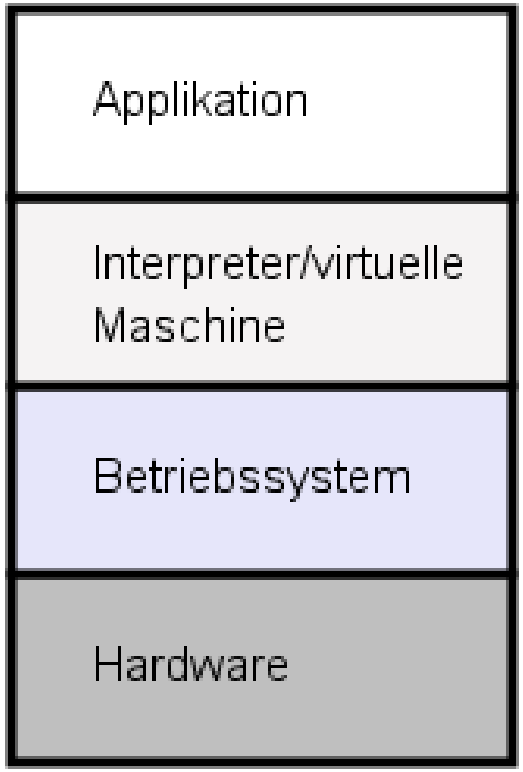


Dynamische digitale Objekte

- Nicht mehr trivial migrierbar
- Andere Strategie: **Emulation**
  - setzt nicht am Objekt selbst an, wie Migration
  - stattdessen wird notwendige Ablaufumgebung des Objekts erhalten
  - Vorteil: Objekt kann unverändert bleiben
- Emulation kein neues Konzept, sondern schon länger in anderen Bereichen genutzt

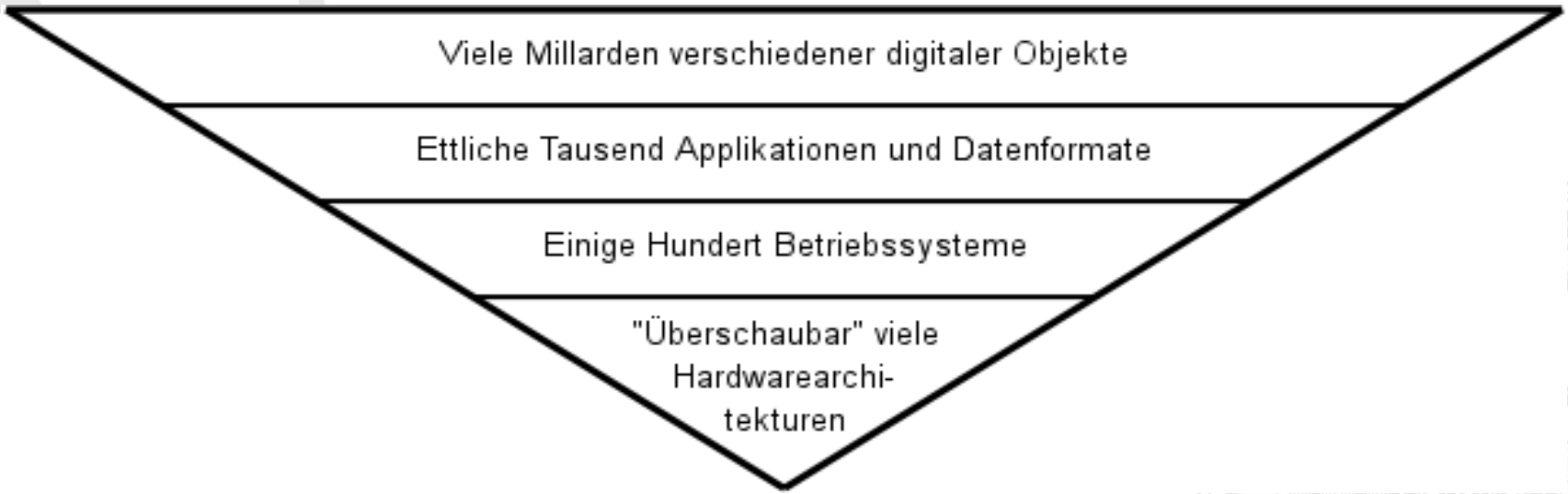
Emulation: Ansatzpunkte I

- verschiedene Ebenen im Software/Hardware Stack
  - Applikation
  - Interpreter, Skriptsprachen (Perl, Python, Ruby, ...), JVM usw.
  - Betriebssystem
  - Hardware



Emulation: Ansatzpunkte II

- Überlegung – Anzahl der notwendigen Emulatoren
- Ebene der Hardware attraktiv



Klassen von Hardware

- Grobe zeitliche Anordnung, andere Kategorisierungen denkbar
  - Mainframes
  - Home-Computer, Arcade, Videospiele
  - Verschiedene Applesysteme, Unix Workstations
  - X86 vom 8bit 8086 bis zu 64bit AMD und Intel CPUs
  - diverse Game-Konsolen und eingebettete Systeme (PDA, Mobiltelefone, ...)

Hardware Emulatoren - Beispiele

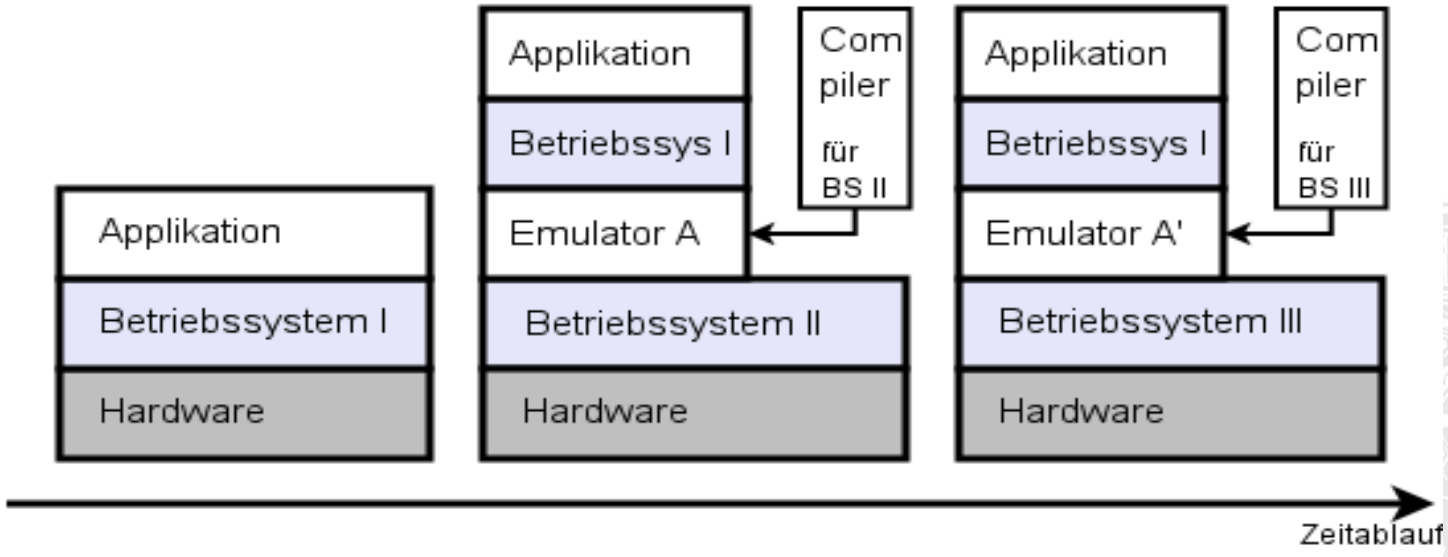
- Mainframes – Computer History Simulation Project, Hercules
- Home-Computer, Arcade – MESS/MAME, Hatari, UAE, Jasper, JaC64, Aranym, Arnold, PCSX, GNUBOY, ...
- Apple – Basilisk (II), Pear PC, QEMU, Mini vMAC
- X86 – QEMU, Bochs, Dioscuri, JPC, *VirtualBox, VMware, Virtual PC, Parallels*, ...
- PDA, Embedded – diverse Entwicklungsplatt-formen der Hersteller, Pose, ...

Emulation – Herausforderungen

- Emulatoren – Brückenfunktion zwischen alter und aktueller Umgebung
- selbst wieder Software
  - stellen Anforderungen an ihre Umgebung als Applikation auf OS/Hardware
  - Übersetzung von Ein- und Ausgaben
  - Datenaustausch zwischen “innen” und “außen”
- regelmäßige Updates unumgänglich

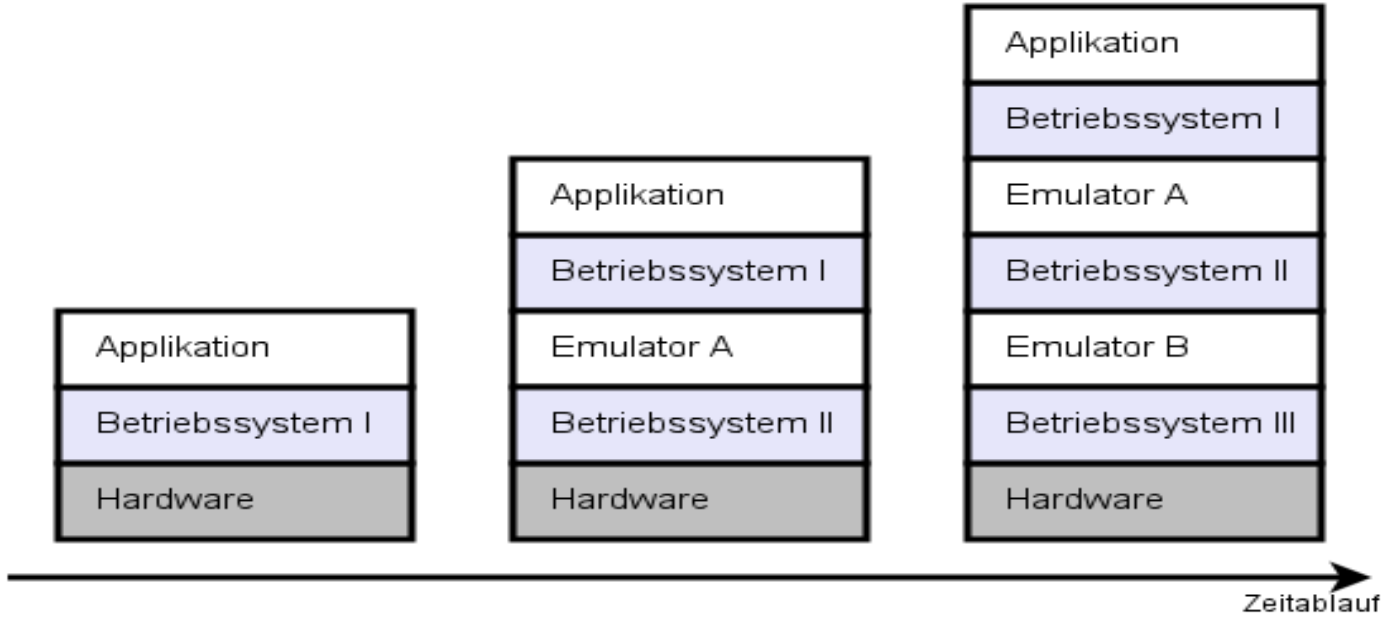
Emulatoren – Langzeitverfügbarkeit I

- Migration – Mitführen des Emulators mit der techn. Entwicklung
  - Anpassung der äußeren Schnittstellen



Emulatoren – Langzeitverfügbarkeit II

- Schachtelung, “Stacking” – gesamte Umgebung des alten Emulators abbilden
  - möglicherweise Transport- und IO-Probleme



Emulatoren – Langzeitverfügbarkeit III

- Universal Virtual Machine (UVC/UVM)
  - abstraktes Komponentensystem
  - fixe Module für Emulation einzelner Hardware-komponenten
  - Beschreibungssprache zur Kombination der Module zu jeweiligem Plattformemulator
- Modulare Emulationsansätze schon im Einsatz/Entwicklung
  - Beispiele: Dioscuri, QEMU, IronBabel

Emulation – Herausforderungen

- keine simple Antwort auf alle Fragen
- Emulator alleine reicht nicht, zusätzlich *digitale Sekundärobjekte*
  - externe, interne Werkzeuge (Hilfsprogramme)
  - Metadaten zur Beschreibung von Emulator und benötigten weiteren Objekten
  - Fonts, Codecs, ...
  - Anleitungen, Handbücher

Bestimmung der Werkzeuganforderungen

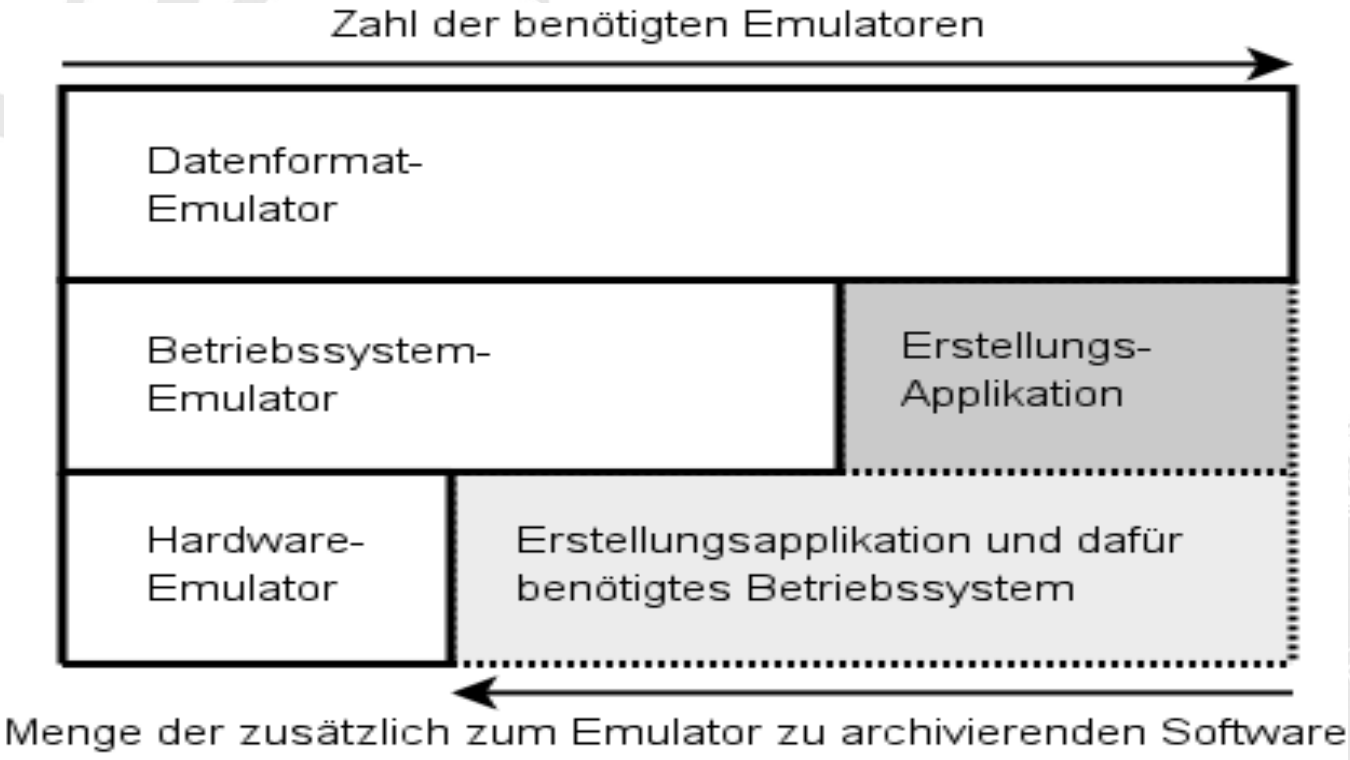
- *View-Path* als Modell zur Ermittlung notwendiger Sekundärobjekte
  - Primärobjekt kann Applikation erfordern
  - Applikation benötigt Ablaufumgebung in Form eines Betriebssystems (typischerweise)
  - Betriebssystem erfordert Hardwareumgebung

Emulation – Softwarearchiv I

- Mehrere View-Path denkbar
  - Verschiedene Applikationen können ein Objekt (unterschiedlich gut) rendern
  - Applikationen können auf unterschiedlichen Betriebssystemen laufen
  - Alte Nutzungsumgebungen können in verschiedenen Emulatoren arbeiten
- Softwarearchiv muss diese Komponenten enthalten
- View-Path aggregierbar

Emulation – Softwarearchiv II

- Werkzeug-Mengen-Matrix

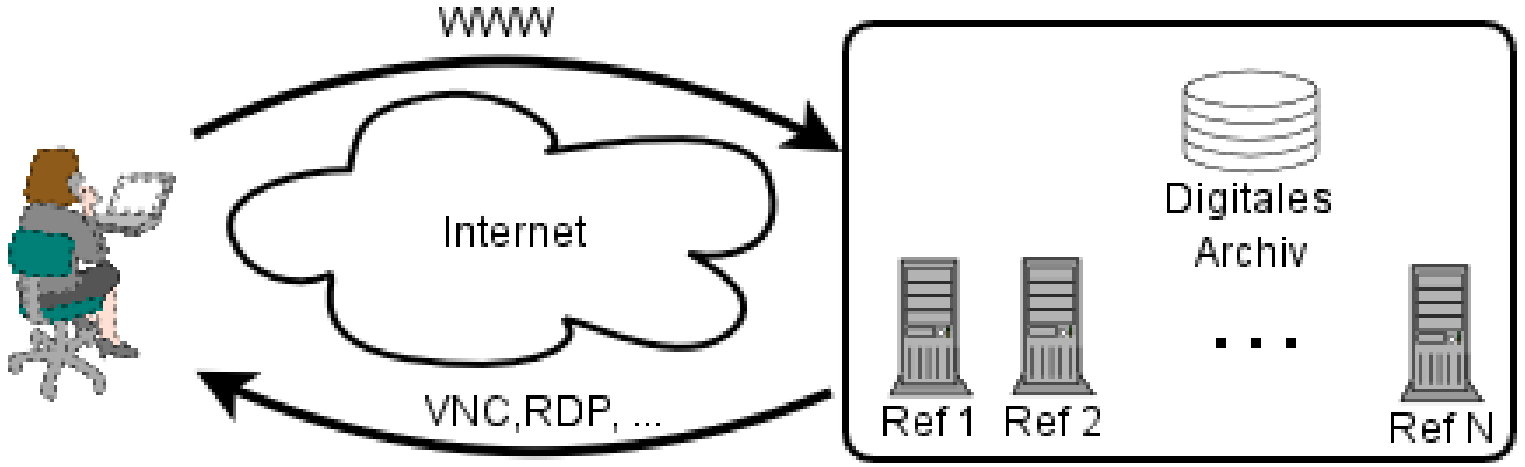


Referenzumgebungen I

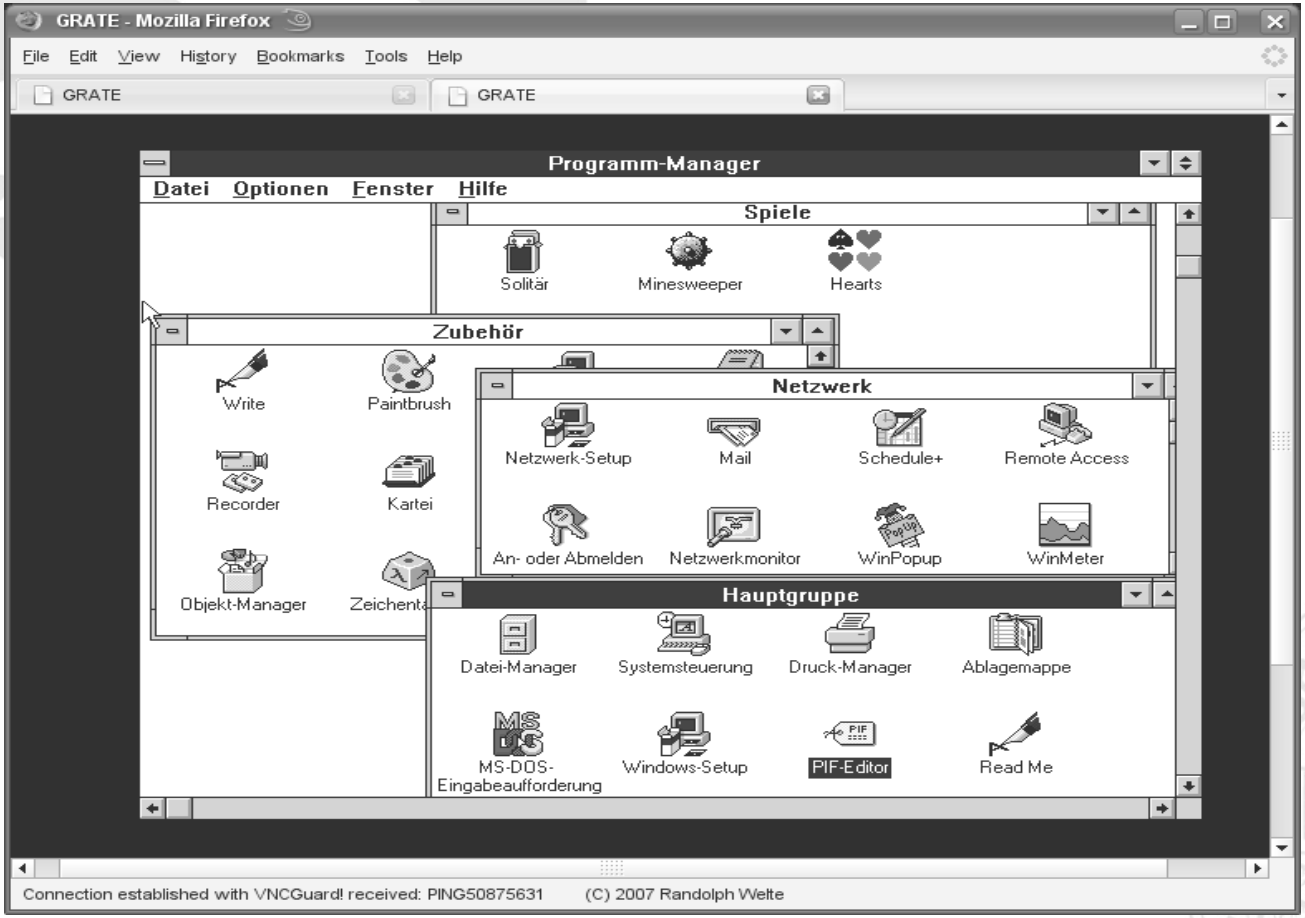
- Endpunkt des jeweiligen View-Path, damit zweiter Bezugspunkt für Emulator
  - Softwareumgebung auf die Emulatoren beständig angepasst/aktualisiert werden müssen
  - Umgebung mit der Endanwender arbeitet
  - veränderlich mit technologischem Fortschritt
- Archivmanagement hält sowohl Software-archiv als auch Referenzumgebungen im Blick

Referenzumgebungen II

- Verschiedene Varianten denkbar
  - Linux-/Windows-Workstations
  - Server für Web-Remote-Services
  - Container für virtualisierte Umgebungen



Referenzumgebungen III



Anwendbarkeit und Grenzen

- Emulation für fast beliebige digitale Objekte geeignet
- Emulation eher für Interaktion, denn automatisierte Barbeitung großer Objektbestände
- Zunehmender Abstand zwischen Ursprungs- und Referenzumgebung
- Auseinanderentwicklung der Bedienungs- und Interaktionsparadigmen
- Ergänzung von Migration, kein Ersatz

ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

550 Jahre Albert-Ludwigs-Universität Freiburg 1457 – 2007

Fragen!? / Kontakt Information

Vielen Dank für die Aufmerksamkeit!

Lehrstuhl für Kommunikationssysteme /
Rechenzentrum der Universität
Herrmann-Herder-Str. 10
79104 Freiburg

Tel. +49 761 203 4698 / 8058
Fax +49 761 203 4640

dsuchod@rz.uni-freiburg.de
rwelte@rz.uni-freiburg.de

www.ks.uni-freiburg.de